Linearly Scaling 3D Fragment Method for Large-Scale Electronic Structure Calculations

Lin-Wang Wang, Byounghak Lee, Hongzhang Shan, Zhengji Zhao,
Juan Meza, Erich Strohmaier, David H. Bailey
Lawrence Berkeley National Laboratory
Berkeley, CA 94720 USA
Email: LWWang@lbl.gov

***Abstract*—**
We present a new linearly scaling three-dimensional fragment (LS3DF) method for large scale *ab initio* electronic structure calculations. LS3DF is based on a divide-and-conquer approach, which incorporates a novel patching scheme that effectively cancels out the artificial boundary effects due to the subdivision of the system. As a consequence, the LS3DF program yields essentially the same results as direct density functional theory (DFT) calculations. The fragments of the LS3DF algorithm can be calculated separately with different groups of processors. This leads to almost perfect parallelization on over one hundred thousand processors. After code optimization, we were able to achieve 60.3 Tflop/s, which is 23.4% of the theoretical peak speed on 30,720 Cray XT4 processor cores. In a separate run on a BlueGene/P system, we achieved 107.5 Tflop/s on 131,072 cores, or 24.2% of peak. Our 13,824-atom ZnTeO alloy calculation runs 400 times faster than a direct DFT calculation, even presuming that the direct DFT calculation can scale well up to 17,280 processor cores. These results demonstrate the applicability of the LS3DF method to material simulations, the advantage of using linearly scaling algorithms over conventional $O(N^3)$ methods, and the potential for petascale computation using the LS3DF method.

I. Introduction

There are many material science and nanoscience problems involving thousands to tens of thousands of atoms that can be accurately simulated only by *ab initio* self-consistent methods. These include nanostructures such as quantum dots and wires, core/shell nanostructures, as well as more conventional systems such as semiconductor alloys. For example, materials that have separate electron states within the energy band gap (mid-band-gap states) have been proposed as next-generation solar cells [1]. Such systems could increase theoretical solar cell efficiencies from 40% to 63% [1]. One potential way to produce mid-band-gap state materials is to use substitutional semiconductor alloys such as $ZnTe_{1-x}O_x$. For a small alloy percentage, ($\approx 3\%$), the oxygen states will repulse the conduction band minimum (CBM) states of the ZnTe and form a separate band inside the energy band gap of ZnTe. Preliminary experimental results have shown that such mid-band-gap states do exist [2]. But the characteristics of these mid-band-gap states are not known (i.e., whether they are spatially isolated or extended), nor is it known whether there is a clear energy band gap between the oxygen induced state and the CBM of ZnTe. If there is no gap, the electron from the ZnTe conduction band will be relaxed into the mid-band-gap states through phonon emission, which will render the material unusable for solar cell applications.

The above questions can be answered by *ab initio* density functional theory (DFT) calculations [3], [4]. However, due to the small percentage of the oxygen atoms, large supercells containing thousands of atoms must be used to describe the random distribution of these oxygen atoms properly. This makes calculations using a direct DFT method impractical. For example, using any of the $O(N^3)$ (here N refers to the system size in atoms) planewave DFT codes (e.g., the Gordon Bell winning Qbox code [5], the similar norm conserving pseudopotential codes PARATEC [6] and PEtot [7], or the widely used ultrasoft pseudopotential code VASP [8]) to simulate the 13,824-atom system discussed in this paper would require between 4-6 weeks of run time for a fully converged self-consistent result using 20,000 cores (processors). This reckoning generously presumes that these codes achieve a high fraction of peak and scale perfectly in performance to 20,000 cores for a single k-point calculation. In reality, of course, performance scaling on such large numbers of cores is usually less than perfect, so that more than 4-6 weeks would likely be required.

In order to solve such problems and to take full advantage of future computer systems now on the drawing boards that will employ hundreds of thousands of cores, we have developed a new linearly scaling method that also exhibits excellent parallel scalability. For the 13,824-atom system mentioned above, this method is roughly 400 times faster than

conventional $O(N^3)$ methods (e.g., Qbox, PARATEC, PEtot, and VASP), yet it yields essentially the same numerical results as the conventional methods. In addition, since it uses a divide-and-conquer scheme, our method is very well-suited for large-scale highly parallel computers.

As will be shown in this paper, the performance of the algorithm scales almost perfectly to 17,280 cores of the Cray XT4 system (Franklin) at the NERSC facility at Lawrence Berkeley National Laboratory, to 30,720 cores of a larger Cray XT4 system (Jaguar) at the NCCS facility at Oak Ridge National Laboratory, and to 131,072 cores of the BlueGene/P system (Intrepid) at the ALCF at Argonne National Laboratory. Based on our performance analysis, LS3DF will scale to a much larger number of cores without any substantive algorithmic obstacles. On the 131,072 core test run we describe below, our code achieved 107.5 Tflop/s or 24.2% of the peak floating-point performance of the Intrepid machine. To our knowledge, this makes LS3DF the first variationally accurate linearly scaling *ab initio* electronic structure code that has been efficiently parallelized to such a large number of processors.

II. Related Work

During the last 15 years there have been numerous developments in linearly scaling *ab initio* methods [9]. Most of these methods use localized orbitals, and minimize the total energy as a function of these orbitals. Unfortunately, the use of localized orbitals can introduce extraneous local minima in the total energy functional, which makes the total energy minimization difficult. On the computational side, such schemes are not well-suited for systems with thousands of cores, because localized orbitals can have strong overlaps which make large-scale parallelization a nontrivial task. As a result of these challenges and in spite of more than a decade of intense research, no one has yet demonstrated an accurate linearly scaling code that can be efficiently used on thousands of cores. Another $O(N)$ approach, the Gordon Bell winning locally self-consistent muliple scattering (LSMS) method [10], has been shown to scale to thousands of cores. The LSMS method is based on the observation that a good approximation to the density can be made by considering only the electronic multiple scattering processes in a region centered about an atom. This observation can then be used to reduce the original problem to one of calculating a single particle Green's function at one atom with an average Green's function to represent the surrounding atoms. This approximation is more applicable to metallic systems than to semiconductors, and the LSMS has only be used to study metallic systems so far. Although there are some codes like SIESTA [11], [12] that can be used to calculate 1000-atom semiconductor systems using linearly scaling algorithms, the performance rates of these codes do not scale well to thousands of cores. In addition, SIESTA is based on simple atomic orbital basis sets that are less accurate than the planewave basis set that we are using here. The divide-and-conquer approach was first proposed by W. Yang [13] and has been used for large system calculations [14]. Based on a different partition scheme, these earlier works did not have the variational principles and boundary effect cancellations of our method. As a result, they are less accurate than our scheme when compared with direct DFT calculations.

III. LS3DF Algorithm Description

A divide-and-conquer scheme is a natural approach for mapping the physical locality of a large problem to the architectural locality of a massively parallel computer. Our method is based on the observation that the total energy of a given system can be broken down into two parts: the electrostatic energy and the quantum mechanical energy (e.g, the kinetic energy and exchange correlation energy). While the electrostatic energy is long-range and must be solved via a global Poisson equation, the computationally expensive quantum mechanical energy is short-range [15] and can be solved locally. The idea is to divide the whole system into small fragments (pieces), calculate the quantum mechanical energies of these fragments, and then combine the separate fragment energies to obtain the energy of the whole system.

A critical issue in a divide-and-conquer scheme such as this is how to combine (patch) the fragments. The core of our algorithm is a novel patching scheme that cancels out the artificial boundary effects caused by the division of the system into smaller fragments. As a result of this cancellation, our results are essentially the same as a direct calculation on the large system, which typically scales as $O(N^3)$, where N is the size of the system in atoms. In our method, once the fragment sizes are chosen to obtain a given numerical accuracy, the computational cost is proportional to the number of fragments. Hence, we call our method the linearly scaling three-dimensional fragment (LS3DF) method. Using a small group of cores to solve the quantum mechanical part of each fragment independently, our method also scales in performance almost perfectly with the number of cores. Only a small overhead is needed to patch the fragment charge densities into a global charge density, to solve the Poisson equation for the whole system, and to divide the global potential

into fragment potentials. As a result, our method can be employed on computer systems with hundreds of thousands of cores.

Our divide-and-conquer scheme is illustrated in Figure 1, which uses a two-dimensional system for clarity. In Figure 1, a periodic supercell is divided into $m_1 \times m_2$ small pieces. From each grid corner (i, j) we can define four fragments, with their sizes S equal to (in units of the smallest piece): $S = 1 \times 1$, 1×2, 2×1 and 2×2, respectively. Suppose we calculate the quantum energy $E_{i,j,S}$ and charge density $\rho_{i,j,S}$ of all of these fragments. Then the total quantum energy of the system can be calculated as $E = \sum_{i,j,S} \alpha_S E_{i,j,S}$, and the total charge density as $\rho(r) = \sum_{i,j,S} \alpha_S \rho_{i,j,S}(r)$. Here $\alpha_S = 1$ for the $S = 1 \times 1$ and 2×2 fragments, and $\alpha_S = -1$ for the $S = 1 \times 2$ and 2×1 fragments. By allowing the usage of both positive and negative fragments in the above summation, the edge and corner effects between different fragments are canceled out, while one copy at the interior region of the fragment will be left to describe the original large system. This scheme can be extended to three dimensions in a straightforward way. The details of this method, as well as some of its novel features, are described in [16], [17].

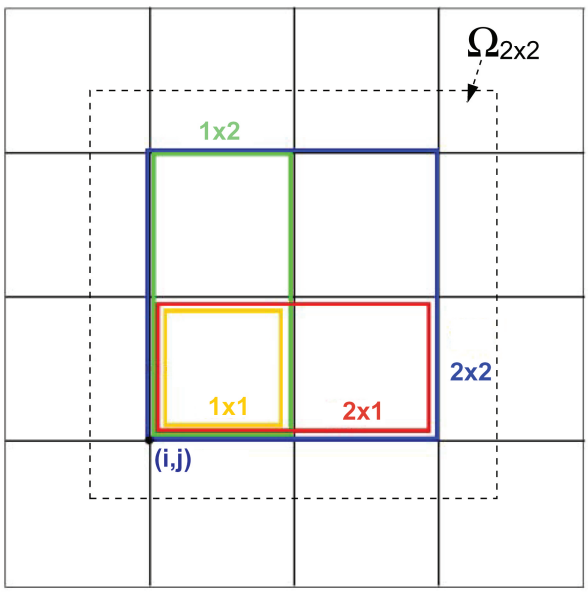

Fig. 1. The division of space and fragment pieces from corner (i, j)

In our implementation of the LS3DF method, we start with a 3D periodic supercell, and divide it into an $M = m_1 \times m_2 \times m_3$ grid. The atoms are assigned to fragments depending on their spatial locations. The artificially created surfaces of the fragments are passivated with hydrogen or partially charged pseudo-hydrogen atoms to fill the dangling bonds [18]. The wavefunctions of the fragments are described by planewaves within a periodic fragment box Ω_F (which is the square region plus a buffer region as shown by the dashed line in Figure 1 for a 2×2 fragment). Norm conserving pseudopotentials are used to describe the Hamiltonian. We use the all-band conjugate gradient method to solve the fragment wavefunctions [7].

Our LS3DF method involves four important steps within each total potential self-consistent iteration, as illustrated in Figure 2. First, a total input potential $V_{in}^{tot}(r)$ (for the whole system) is provided. Secondly, the Gen_VF routine generates for each fragment F, the potential $V_F(r) = V_{in}^{tot}(r) + \Delta V_F(r)$, $r \in \Omega_F$, where $\Delta V_F(r)$ is a fixed passivation potential for each fragment F which is only nonzero near its boundary [16]. Note that $V_F(r)$ is only defined in Ω_F. Third, PEtot_F solves Schrödinger's equation on each fragment for its wavefunctions $\psi_i^F(r)$. After the fragment wavefunctions $\psi_i^F(r)$ are solved for, the fragment charge density is computed, $\rho_F(r) = \sum_i |\psi_i^F(r)|^2$, and the charge density for the overall system is patched together $\rho_{tot}(r) = \sum_F \alpha_F \rho_F(r)$ by subroutine Gen_dens. In the final step carried out by subroutine GENPOT, a global Poisson equation is solved using FFTs to obtain the global potential $V_{out}^{tot}(r)$. After potential mixing from previous iterations, the modified $V_{out}^{tot}(r)$ is used as the input for the next self-consistent iteration. Self-consistency is reached as $V_{out}^{tot}(r)$ approaches $V_{in}^{tot}(r)$ within a specified tolerance.

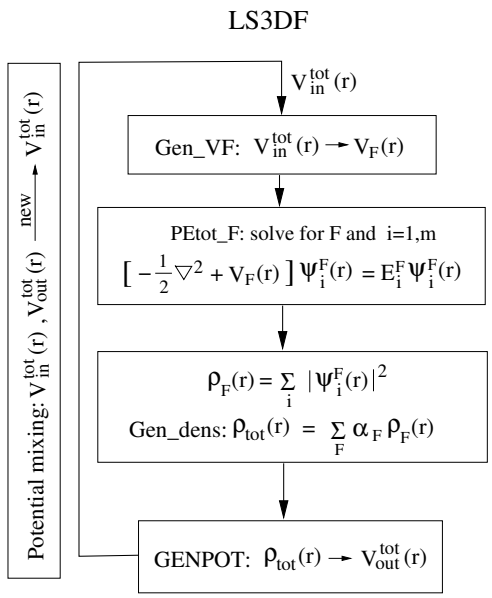

Fig. 2. LS3DF flow chart.

IV. LS3DF Code Optimizations

LS3DF was developed specifically for large-scale parallel computation over the last one and one-half years, although it builds on the previous $O(N^3)$ DFT code PEtot [7] that has been in development for nearly ten years. The four major subroutines in LS3DF, namely Gen_VF, PEtot_F, Gen_dens, and GENPOT (Figure 2) were developed in steps. Initially, as a proof of concept, they were developed as separate executables using file I/O to pass the data between them. Later, these subroutines were integrated into a single executable. These codes are written in Fortran-90, using MPI for parallel computation.

In the most recent phase of the development, which yielded the code that we used in this study, we first focused on the PEtot_F subroutine, which dominates run time. It ran at about 15% of peak performance in the earlier phases of development. PEtot_F is derived from the PEtot code, which uses the same planewave q-space parallelization as other standard codes, such as PARATEC and Qbox. However, to save on memory requirements, it used a band-by-band algorithm that contributed to its relative low performance rate. Detailed profiling and analyses were carried out to increase the performance of PEtot_F. Closer analysis revealed that since PEtot_F solved for one electron wavefunction at a time (band-by-band), the majority of operations (for nonlocal pseudopotential and wavefunction orthogonalization) were performed using BLAS-2 routines. The file I/O for communication also took a substantial amount of time.

Based on these analyses, we performed four major code optimizations to improve its performance and scalability:

1) Within PEtot_F, solve for all the electron wavefunctions simultaneously (i.e., the all-band scheme), instead of one wavefunction at a time (i.e., the band-by-band scheme). With this approach, we can utilize BLAS-3 operations such as DGEMM, yielding higher performance than is possible with BLAS-2 routines. A typical matrix size for one of our fragments would be 3000×200.
2) Implement a different algorithm for data communication between cores in Gen_VF and Gen_dens to allow better scaling for large numbers of cores.
3) Store the data in memory (through an LS3DF global module), and communicate with MPI calls, rather than store the data on disk and communicating via file I/O as in the earlier versions of the code. This change has resulted in a major improvement in scalability and overall performance.
4) Eliminate the setup overhead during each self-consistent call for PEtot_F and GENPOT through storage of the relevant variables in the LS3DF global module.

With regards to the first item, the original PEtot_F band-by-band algorithm has the advantage of requiring a rather modest amount of memory. But on the Cray XT4 system, with 2 GBbyte memory per core, we have enough memory to use the all-band method, which involves matrix-matrix multiplication that can be performed using BLAS-3 routines. This all-band optimization required changes in the data layout and rearrangements of the loop structures. We have also implemented a new orthogonalization scheme for the all-band method. Instead of imposing the orthonormal condition for the wavefunction using a Gram-Schmidt scheme at each conjugate gradient step, we only impose the orthonormal condition after a few conjugate gradient steps by calculating an overlapping matrix. The use of the overlapping matrix instead of the direct band-by-band Gram-Schmidt algorithm also permits the use of BLAS-3 library routines such as DGEMM.

In the wake of this code optimization, the performance of the stand-alone PEtot code has increased from 15% of the theoretical peak to 56% for large system calculations. This performance ratio is close to that of the best planewave codes, such as PARATEC and Qbox. The performance ratio of PEtot_F for our largest fragments is 45% on Franklin, which is slightly slower than the stand-alone code, probably due to the small size of the fragment. In a 2000-atom CdSe quantum rod sample problem, after the final code optimization, for 8,000-core runs, the execution times for the four subroutines of the code have been reduced to: Gen_VF 2.5 seconds (from the original 22 seconds), PEtot_F 60 seconds (from the original 170 seconds), Gen_dens 2.2 seconds (from the original 19 seconds), and GENPOT 0.4 seconds (from the original 22 seconds). These timings represent a factor of four overall improvement compared with the previous version. For Gen_VF and Gen_dens, the improvement is a factor of 10, while for GENPOT, the improvement is a factor of 50. The improvements for these three subroutines are critical for large-scale runs.

Shortly before the final completion of this study, we were able to obtain access to the Intrepid system, which is a large BlueGene/P system in the ALCF facility at Argonne National Laboratory. For our largest run on the Intrepid system (using 131,072 cores), we further improved Gen_VF and Gen_dens routines by employing point-to-point isend and ireceive operations. As a result, on our Intrepid runs,

these two routines together comprised less than 2% of the total run time. In particular, the breakdown for one self-consistent field (SCF) iteration is as follows: Gen_VF (0.37 sec.), PEtot_F (54.84 sec.), Gen_dens (0.56 sec.) and GENPOT (1.23 sec.). Since such a large percentage of the time now goes to PEtot_F (which has no inter-group communication), this bodes well for the future scalability of this code on petascale computer systems.

Although we are pleased with the performance and scalability of the present version, we do plan on making some additional improvements in the months and years ahead including: (1) two-level parallelization in PEtot_F, to achieve greater parallelism; and (2) replacing DGEMM with a custom routine specialized for PEtot_F. The two-level parallelization will include parallelization over the plane wave basis set (as currently implemented) and on the wave function index i as indicated in Figure 2. This parallelization will allow us to increase the number of processors, N_p, per group, thereby increasing the scalability of our code even further, especially for strong scaling.

V. Test Systems

In order to test the scaling and flop/s performance of the LS3DF code, we set up a series of test problems involving $ZnTe_{1-x}O_x$ alloy systems. These alloy systems are in a distorted zinc blende crystal structure, with 3% of Te atoms being replaced by oxygen atoms. Although the LS3DF method can be used to calculate the force and relax the atomic position, for these particular systems we found that the atomic relaxation can be described accurately by the classical valence force field (VFF) method [19]. Here we are more interested in the charge density and electronic structure for a given atomic configuration which is relaxed using VFF. These alloy systems are characterized by the sizes of their periodic supercells. The size of a supercell can be described as $m_1 \times m_2 \times m_3$ in the unit of the cubic eight-atom zinc blende unit cell. Thus, the total number of atoms is equal to $8m_1m_2m_3$. We used a nonlocal norm-conserving pseudopotential to describe the Hamiltonian, and we used a planewave basis function with a 50 Ryd energy cutoff to describe the wavefunction. The real space grid for each eight-atom unit cell is $40 \times 40 \times 40$. The d-state electrons in Zn atom are not included in the valence electron calculation. Thus in average, there are four valence electrons per atom.

We found that for our fragment calculations, a reciprocal q-space implementation of the nonlocal potential is faster than a real-space implementation. Thus, we have used a q-space nonlocal Kleinman-Bylander projector for the nonlocal potential calculation [20]. The accuracy of LS3DF, as compared with the equivalent DFT computation, increases exponentially with the fragment size. For the LS3DF calculation, we have used the eight-atom cubic cell as our smallest fragment size, as shown in Figure 1. Using this fragment size, the LS3DF results are very close to direct DFT calculated results. For example, the total energy differed by only a few meV per atom, and the atomic forces differed by 10^{-5} a.u. [16]. For all practical applications, this means the LS3DF and the direct DFT results are essentially the same.

To test the weak scaling of the LS3DF code, we have chosen alloy supercells of dimensions $m_1 \times m_2 \times m_3$, namely $3 \times 3 \times 3$, $4 \times 4 \times 4$, $5 \times 5 \times 5$, $6 \times 6 \times 6$, $8 \times 6 \times 9$, $8 \times 8 \times 8$, $10 \times 10 \times 8$ and $12 \times 12 \times 12$. These problems correspond to 216, 512, 1000, 1728, 3456, 4096, 6400, and 13824 atoms, respectively. To study the physics of the oxygen induced states, large supercells are needed to properly describe the atomic configuration due to the small oxygen percentages (e.g., 3%) used in laboratory experiments.

For the $3 \times 3 \times 3$ system, we have also calculated the full system with a direct local density approximation (LDA) method (using PEtot). The band gap and eigenenergy differences between the direct LDA method and the LS3DF method are about 2 meV (for the LS3DF method, we took its converged potential, then calculated the eigenenergy of the full system). Since the energy gaps we wish to investigate are around a few tenths of an eV to a few eV, the LS3DF method is extremely accurate for our study. In fact, it is numerically accurate enough for almost all material science simulations in terms of reproducing the direct LDA results. For example, in a previous study we used LS3DF to calculate thousand-atom quantum rods and their dipole moments [16]. These calculated dipole moments differed from the direct LDA results by less than 1%.

For the runs we did on the Intrepid system, we modified the above parameters as follows. First of all, we employed a cutoff value of 40 Ryd instead of 50, and we employed a real-space grid of size $32 \times 32 \times 32$. These changes were made to adjust for the smaller amount of main memory per core on the BlueGene/P system. Secondly, we generated larger problem sizes, including alloy supercells of dimensions $4 \times 4 \times 4$, $8 \times 4 \times 4$, $8 \times 8 \times 4$, $8 \times 8 \times 8$, $16 \times 8 \times 8$ and $16 \times 16 \times 8$.

VI. Computational Results

To assess the optimizations implemented in LS3DF and to demonstrate the benefits of this new code, we

conducted several computational experiments. We executed LS3DF with a constant problem size across the currently available range of concurrency (i.e., "strong scaling"), and for a variety of problem sizes (i.e., "weak scaling"). We then compared code performance to other important DFT codes such as PARATEC and VASP.

Part of these benchmark runs were performed on the Franklin system at NERSC. This is a Cray XT4 system with 9,660 compute nodes, each of which has two 2.6 GHz AMD Opteron cores and 4 GByte main memory. The entire system has a theoretical peak performance rate of 101.5 Tflop/s. The second set of runs were performed on the Jaguar system at NCCS. Jaguar has 7,832 XT4 compute nodes, each with a quad-core 2.1 GHz AMD Opteron processor and 8 GByte of memory. The theoretical peak performance rate of this system is approximately 263 Tflop/s. The third set of runs were performed on the Intrepid system at ALCF. This is a BlueGene/P system with 40,960 nodes and 163,840 cores, and a theoretical peak performance rate of 556 Tflop/s.

A summary of our performance results is given in Table I. Figures are listed in separate sections of the table for runs on Franklin, Jaguar and Intrepid. Tflop/s figures in the table for various problem sizes were calculated based on operation counts measured on the Franklin system using the CrayPat tool [21]. For the very largest problems, which we were unable to run on Franklin, we estimated the operation count based on the number of fragments and the small-problem operation counts. This estimation scheme was found to be consistently within 1% of the actual operation count for those problems that we could run on Franklin. Note that our Tflop/s figures reported in Table 1 were calculated based on wall-clock time. On Franklin, these figures are roughly 10% lower than the Tflop/s figures reported by the CrayPat tool, which uses user time rather than wall-clock time, but we decided to use the wall-clock reckoning for consistency across all systems.

To evaluate the strong scaling behavior of LS3DF, we chose a medium-sized problem with 3,456 atoms and a fragment grid size of $m_1 \times m_2 \times m_3 = 8 \times 6 \times 9$. In this test, we used $N_p = 40$, where N_p is the number of cores within each group used in PEtot_F. The value of 40 was determined by the parallel efficiency for each group. Our experience has shown (for example, Table I, third test case for the Jaguar system) that when the value of N_p is increased beyond 40, the scaling within each group drops off, which drives the overall efficiency down. We increased the number of groups N_g from 27 to 432. This change represents a 16-fold range of concurrency levels from 1,080 cores to 17,280 cores on the Franklin system. To estimate the run time, we executed two SCF iterations of LS3DF and analyzed the times for the second iteration, since this is the iteration which will be iterated several dozen times for a converged calculation. The first iteration has some small additional overhead (due to array and index setups), while subsequent iterations behave very similarly to each other. We confirmed this behavior during our full scientific runs with 60 SCF iterations.

sys. size	atoms	cores	N_p	Tflop/s	% peak
Franklin					
$3 \times 3 \times 3$	216	270	10	0.57	40.4%
$3 \times 3 \times 3$	216	540	20	1.14	40.8%
$3 \times 3 \times 3$	216	1080	40	2.27	40.5%
$4 \times 4 \times 4$	512	1280	20	2.64	39.6%
$5 \times 5 \times 5$	1000	2500	20	5.15	39.6%
$6 \times 6 \times 6$	1728	4320	20	8.72	38.8%
$8 \times 6 \times 9$	3456	1080	40	2.28	40.5%
$8 \times 6 \times 9$	3456	2160	40	4.51	40.2%
$8 \times 6 \times 9$	3456	4320	40	8.88	39.5%
$8 \times 6 \times 9$	3456	8640	40	17.04	37.9%
$8 \times 6 \times 9$	3456	17280	40	31.35	34.9%
$8 \times 8 \times 8$	4096	2560	20	5.46	41.0%
$8 \times 8 \times 8$	4096	10240	20	19.72	37.0%
$10 \times 10 \times 8$	6400	2000	20	4.18	40.2%
$10 \times 10 \times 8$	6400	16000	20	29.52	35.5%
$12 \times 12 \times 12$	13824	17280	10	32.17	35.8%
Jaguar					
$8 \times 8 \times 6$	3072	7680	20	17.3	26.8%
$8 \times 8 \times 6$	3072	15360	40	33.0	25.6%
$8 \times 8 \times 6$	3072	30720	80	53.8	20.9%
$8 \times 6 \times 9$	3456	17280	40	36.5	25.2%
$16 \times 8 \times 6$	6144	15360	20	33.6	26.0%
$16 \times 12 \times 8$	12288	30720	20	60.3	23.4%
Intrepid					
$4 \times 4 \times 4$	512	4096	64	4.4	31.6%
$8 \times 4 \times 4$	1024	8192	64	8.8	31.5%
$8 \times 8 \times 4$	2048	16384	64	17.5	31.4%
$8 \times 8 \times 8$	4096	32768	64	34.5	31.1%
$16 \times 8 \times 8$	8192	65536	64	60.2	27.1%
$16 \times 16 \times 8$	16384	131072	64	107.5	24.2%

TABLE I
SUMMARY OF TEST RESULTS. "% PEAK" IS THE FRACTION OF THE PEAK PERFORMANCE FOR THE NUMBER OF CORES USED.

Figure 3 shows the speedup of LS3DF and the PEtot_F component for the range of cores evaluated on Franklin. Speedup and parallel efficiency figures for the 17,280-core runs (using the 1,080-core run as baseline), were 15.3 and 95.8% for the PEtot_F portion, and 13.8 and 86.3% for LS3DF, both of which

are excellent. Overall, LS3DF achieved a performance rate of 31.35 Tflop/s on 17,280 cores. All computations are performed on 64-bit floating-point data.

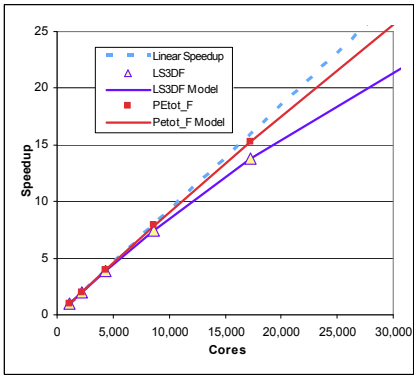

Fig. 3. Strong scaling speedups for LS3DF and PEtot_F. The curves are models based on Amdahl's Law.

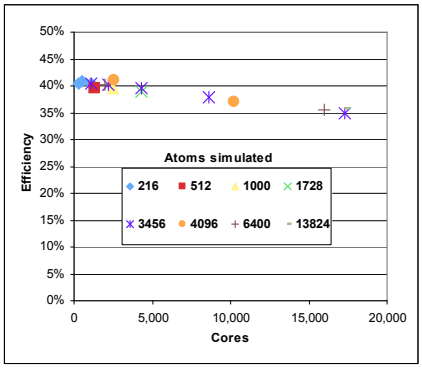

Fig. 4. Computational efficiency for different runs on Franklin

We analyzed the results of our strong scaling experiment with Amdahl's Law:

$$P_p = P_s \left(\frac{n}{1 + (n-1)\alpha} \right). \quad (1)$$

Here P_p is parallel performance, P_s is the serial performance, n is the number of cores, and α is the fraction of serial work in the code. In particular, we employed least-squares fitting to determine the parameters P_s and α. The resulting formula fits our performance data extremely well, with an average absolute relative deviation of the fitting, namely $\sum_n |(P_{fitted}/P_{measured} - 1)| / n$, of only 0.26% and a single maximal deviation of 0.48%. Fitted values for the single core performance are 2.39 GFlop/s for the effective single core performance and hypothetical fractions of the remaining serial work components of 1/362,000 for PEtot_F and 1/101,000 overall for LS3DF.

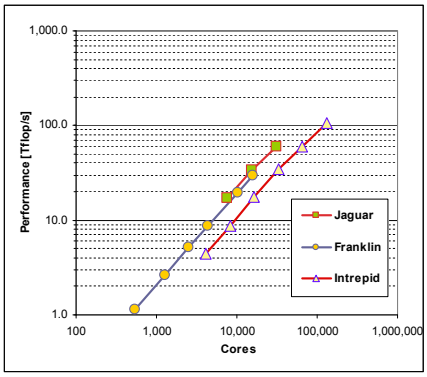

Fig. 5. Weak scaling floating point operation rates on different machines.

In Figure 4, we show computational efficiencies for a variety of problems and different code execution parameters for runs on Franklin. Overall, the excellent scalability demonstrated in our strong scaling experiment is confirmed. The small variations in code performance for a given concurrency level appear to depend, to first order, on code execution parameters such as the size N_p of processor groups working on individual fragments. We also notice that for a given concurrency, the computational efficiency is almost independent of the size of the physical system studied. The slight drop of efficiency for very high concurrency is again mostly due to Gen_VF and Gen_dens. This drop-off should be reduced if we ran our latest version of the code on Franklin and Jaguar, which employs a faster communication scheme for these two routines. For the same reason, the LS3DF curve in Figure 3 would become closer to the PEtot_F curve should the latest version of the code be used on Franklin.

Figure 5 shows the total Flop/s rate for weak scaling runs (with constant number of atoms to number of cores ratio) on different machines. These data reflect the fact that Jaguar has the faster per processor speed, while Intrepid has the largest number of processors and thus yields the largest total performance rate (107 Tflop/s). The fairly straight lines suggest that the code is well poised for future petascale computer systems.

On our sustained test run on Franklin, we achieved convergence on the $8 \times 6 \times 9$ system with 60 iterations, using 17,280 cores of Franklin. This run required one hour of run time, with one minute per iteration. The

total sustained performance rate is 31.4 Tflop/s, which is 35% of the theoretical peak rate (89.9 Tflop/s) on 17,280 cores.

To demonstrate the advantages of LS3DF over a conventional $O(N^3)$ method, we also performed comparable calculations on Franklin with commonly used DFT codes, including VASP and PARATEC, which have SCF convergence rates similar to that of LS3DF [16]. We calculated the $3 \times 3 \times 3$ and $4 \times 4 \times 4$ systems with PARATEC, stand-alone PEtot and VASP. The performance rates of PARATEC and stand-alone PEtot are within 5% of each other. From the execution times for these two systems, we can see that the $O(N^3)$ regime is already reached by the $4 \times 4 \times 4$ system. For PARATEC, we used the same pseudopotentials, energy cutoff, and number of conjugate gradient steps for each SCF iteration, as in LS3DF. PARATEC required 340 seconds for one SCF iteration using 320 cores. For VASP, one iteration required about 200 seconds. However, a direct comparison with VASP is clouded by the fact that it uses different pseudopotential and planewave cutoff values, and it takes fewer conjugate gradient (or residual minimization) steps per SCF iteration. Nevertheless, the key fact here is that PARATEC and VASP have times that are within a factor of two. From the $O(N^3)$ scaling of PARATEC, we deduce that its computation time will cross with the LS3DF time at about 600 atoms. For the 13,824-atom problem we simulated using LS3DF, we estimate PARATEC will be 400 times slower, even under the generous presumption that its performance scales perfectly to 17,280 cores. In summary, while LS3DF requires only three hours to perform a fully converged calculation for such a physical system, the $O(N^3)$ codes would require roughly six weeks, making them impractical for most research purposes. This large ratio (400) will be even more dramatic as we consider runs on even larger physical configurations, e.g., for dislocation or grain boundary problems.

VII. Science Results

As mentioned above, we have achieved fully converged results for the $m_1 \times m_2 \times m_3 = 8 \times 6 \times 9$ system. This physical system has 3,456 atoms, and requires 60 SCF iterations (the outer loop in Figure 2) to achieve full convergence. Using 17,280 processors, the run requires one minute for each iteration, and thus one hour for the entire calculation. The SCF convergence of the system can be measured by the difference between the input $V_{in}(r)$ and output $V_{out}(r)$ potentials, as shown in Figure 2. We have chosen to use the absolute difference rather than the relative difference because $V_{in}(r)$ can have an arbitrary shift, and because the absolute difference can be directly related to the energy difference. The difference $\int |V_{in}(r) - V_{out}(r)| d^3r$ as a function of self-consistent iteration is shown in Figure 6 (in atomic units). As one can see, overall this difference decays steadily. However, there are a few cases where this difference jumps. This is typical in the potential mixing method, since there is no guarantee that this difference will decrease at every step. Overall, the convergence rate is satisfactory, and the final 10^{-2} potential difference is comparable to the criterion used in our nanosystem dipole moment calculations. In addition, because the charge density response to a potential change in LS3DF is similar to a direct LDA method, and we are using the same charge mixing scheme, we would expect that the LS3DF method will have similar convergence properties as the direct LDA method, and should therefore converge for all systems with a band gap.

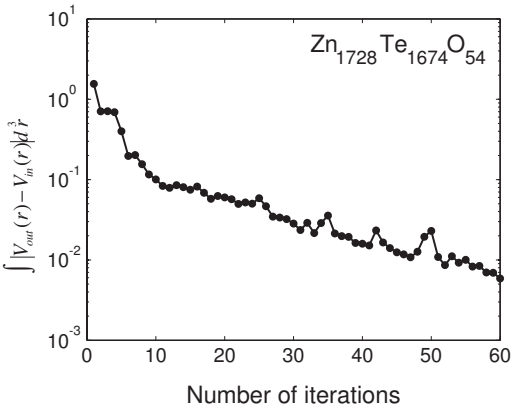

Fig. 6. LS3DF convergence: Input and output potential difference as a function of self-consistent iteration steps.

The converged potential $V(r)$ is then used to solve the Schrödinger equation for the whole system for only the band edge states. This was done using our folded spectrum method (FSM) [22]. Since not all the occupied eigenstates are calculated, the FSM method scales linearly with the size of the system. Overall this step does not take much time and it can be considered as a fast post-process of the LS3DF calculations. There is a well-known LDA band gap error that can be corrected using a modified nonlocal pseudopotential for the s, p, d states [19]. The calculated CBM state is shown in Figure 7(a), while the highest oxygen induced states is shown in Figure 7(b). Between the CBM and the highest oxygen induced state, there is a 0.2 eV band gap. This should prevent the electron in CBM from falling down to the oxygen induced states. Thus, our simulations predict that the $ZnTe_{0.97}O_{0.03}$ alloy could be used for solar cell applications.

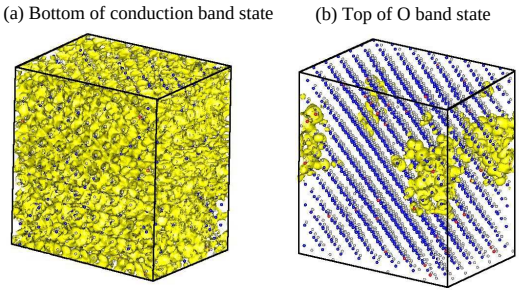

Fig. 7. Isosurface plots (yellow) of the electron wavefunction squares for the bottom of conduction band (a), and top of oxygen-induced band (b). The small grey dots are Zn atoms, the blue dots are Te atoms, and the red dots are oxygen atoms.

One interesting point is that the oxygen induced states form a very broad band (0.7 eV) inside the band gap of ZnTe. As a result, its theoretical maximum efficiency of a solar cell made from this alloy will be smaller than the 63% estimated based on a narrow mid-band-gap [1]. Also, as shown in Figure 7(b), the oxygen induced states can cluster among a few oxygen atoms. Such a clustering is more localized in the high energy states than in the lower energy states within the oxygen induced band, which will significantly reduce the electron mobility (i.e., conductivity) in those states.

VIII. Conclusions

In summary, we have developed and deployed a fundamentally new approach to the problem of large-scale *ab initio* electronic structure calculations. Our approach targets systems with a band gap and that require highly accurate planewave based calculations. It can be applied to nanostructures, defects, dislocations, grain boundaries, alloys and large organic molecules. It simultaneously addresses the two critical issues for any large-scale computer simulation: the scaling of the total computational cost relative to the size of the physical problem, and the parallel scaling of the computation to very large numbers of processor cores. Unlike some other recent Gordon Bell prize papers, our paper represents not merely an adaptation and improvement of an existing code to a newly available large computer system. Instead, we have designed and implemented a new algorithm to solve an existing physical problem with petascale computation in mind from the beginning.

One of our test computations, for the $m_1 \times m_2 \times m_3 = 16 \times 12 \times 8$ ZnTeO system, has 12,288 atoms and would require 60 self-consistent iterations (the outer loop in Figure 2) to achieve full convergence. Using 30,720 cores of the Jaguar system, the run required 115 seconds for each iteration, so that an entire 60-iteration run could be completed in less than two hours. The sustained performance of our test run was 60.3 Tflop/s. In separate test runs on the Intrepid system, on a $16 \times 16 \times 8$ ZnTeO system with 16,384 atoms, we achieved 107.5 Tflop/s on 131,072 cores, so that a full calculation could be completed within one hour. These performance figures (and all others reported in this paper) are for 64-bit floating-point operations.

In another test computation, namely a 13,824-atom ZnTeO alloy calculation, performed on the Franklin system, our code ran roughly 400 times faster than would be possible with any of the other planewave research codes currently in use, mostly because of the linearly scaling algorithm of our method, as compared with the $O(N^3)$ algorithms of most other methods. In short, our code is the first variationally accurate linearly scaling *ab initio* electronic structure code which has been efficiently parallelized to over 100,000 processor cores. Our code makes it possible to tackle problems which were not amenable before.

Our simulation yielded substantive scientific results. First, we found that there is a 0.2 eV band gap between the ZnTe conduction band and the oxygen induced band, implying that this alloy can be used for solar cell applications. Secondly, we found that the oxygen induced states form a very broad band (0.7 eV) inside the band gap of ZnTe. As a result, our simulations predict that the theoretical maximum efficiency of solar cells made from this alloy will most likely be lower than the 63% figure estimated based on a narrow mid-band-gap. Also, as shown in Fig. 7(b), the oxygen induced states can cluster among a few oxygen atoms. Such a clustering will have an impact on the mobility of the electrons in those states.

Lastly, based on our performance analysis, we see no intrinsic obstacle to scaling our code to run on over 1,000,000 processing cores and over 1 Pflop/s performance.

Acknowledgements This work was supported by the Director, Office of Science, Basic Energy Sciences, Division of Material Science and Engineering, and the Advanced Scientific Computing Research Office, of the U.S. Dept. of Energy under Contract No. DE-AC02-05CH11231. It used resources at the National Energy Research Scientific Computing Center (NERSC), the National Center for Computational Sciences (NCCS) and the Advanced Leadership Computing Facility (ALCF).

References

[1] A. Luque and A. Marti. Increasing the efficiency of ideal solar cells by photon induced tansitions at intermediate lavels. *Phys.*

Rev. Lett., 78:5014, 1997.
[2] K.M. Yu, W. Walukiewicz, J. Wu, W. Shan, J.W. Beeman, M.A. Scarpulla, O.D. Dubon, and P. Becta. Diluted ii-vi oxide semiconductors with multiple band gaps. *Phys. Rev. Lett.*, 91:246403, 2003.
[3] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136:B864, 1964.
[4] W. Kohn and L.J. Sham. Self-consistent equations including exchange and correlation effects. *Phys. Rev.*, 140:A1133, 1965.
[5] F. Gygi, E. Draeger, B. R. de Supinski, R. K. Yates, F. Franchetti, S. Kral, J. Lorenz, C. W. Ueberhuber, J. Gunnels, and J. Sexton. Large-scale first-principles molecular dynamics simulations on the bluegene/l platform using the qbox code. *ACM*, 2005.
[6] http://www.nersc.gov/projects/paratec.
[7] L.-W. Wang. Parallel planewave pseudopotential ab initio package, http://hpcrd.lbl.gov/ linwang/petot/petot.html, 2004.
[8] http://cms.mpi.univie.ac.at/vasp/.
[9] G. Goedecker. Linear scaling electronic structure methods. *Rev. Mod. Phys*, 71:1085, 1999.
[10] http://www.psc.edu/general/software/packages/lsms/.
[11] José M Soler, Emilio Artacho, Julian D Gale, Alberto García, Javier Junquera, Pablo Ordejón, and Daniel Sánchez-Portal. The siesta method for ab initio order-n materials simulation. *Journal of Physics: Condensed Matter*, 14(11):2745–2779, 2002.
[12] http://www.uam.es/departamentos/ciencias/fismateriac/siesta.
[13] W. Yang. Direct calculation of electron density in density-functional theory. *Phys. Rev. Lett.*, 66:1438, 1991.
[14] F. Shimojo, R.K. Kalia, A. Nakano, and P. Vashishta. Embedded divide-and-conquer algorithm on hierarchical real-space grids: parallel molecular dynamics simulation based on linear-scaling density functional theory. *Comput. Phys. Commun.*, 167:151, 2005.
[15] W. Kohn. Density functional and density matrix method scaling linearly with the number of atoms. *Phys. Rev. Lett.*, 76(17):3168–3171, 1996.
[16] L.-W. Wang, Z. Zhao, and J. Meza. Linear scaling three-dimensional fragment method for large-scale electronic structure calculations. *Phys. Rev. B*, 77:165113, 2008.
[17] Z. Zhao, J. Meza, and L.-W. Wang. A divide and conquer linear scaling three dimensional fragment method for large scale electronic structure calculations. *J. Phys. Cond. Matter*, 20(294203), 2008.
[18] L.-W. Wang and J. Li. First-principles thousand-atoms quantum dot calculations. *Phys. Rev. B*, 69:153302, 2004.
[19] J. Schrier, D.O. Demchenko, L.W. Wang, and A.P. Alivisatos. Optical properties of zno/zns and zno/znte heterostructures for photovoltaic applications. *NanoLett.*, 7:2377, 2007.
[20] M.C. Payne, M.P. Teter, D.C. Allan, T.A. Arias, and J.D. Joannopoulos. Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients. *Rev. Mod. Phys.*, 64:1045, 1992.
[21] http://docs.cray.com/books/s-2315-50/html-s-2315-50/z1055157958smg.html.
[22] L.W. Wang and A. Zunger. Solving schrodinger's equation around a desired energy: Application to silicon quantum dots. *J. Chem. Phys.*, 100:2394, 1994.